# STUDY OF THE ASSEMBLY MOTION FOR A VIRTUAL ASSEMBLY SYSTEM

**Aurel FRATU*, Laurent VERMEIREN**, Jean-Francois BRETHÉ*****
[*] Transilvania University of Brasov, Romania
[**] Université de Valenciennes, France
[***] Université du Havre, France

**Abstract.** This paper describes the virtual assembly automation systems, as decision support systems, for the better knowledge of assembly operation. The study of this procedure is described as well as the necessity for assembly systems design. The work draws on research into product and manufacturing knowledge models. In this paper, the authors have proposed a virtual prototype model for assembly system architecture of robotic assembly automation.

This paper presents a methodology for solid modeling in a virtual environment that is precisely performed in an intuitive manner through constraint-based manipulations Constraint-based manipulations are realized by allowable motions for precise 3D interactions in the virtual environment. A mathematical matrix is presented for representing allowable motions. A procedure-based degree-of-freedom combination method for 3D constraint solving is presented for deriving the allowable motions from constraints. A prototype system has been implemented to testify the feasibility of the presented methodology.

**Keywords:** virtual robotic assembly systems, solid model representation, constraint-based manipulations

## 1. Introduction

The assembly process with robots systems is faster, more efficient and precise than ever before. Invention of robots has brought about revolutionary changes in the field of industrial manufacturing. Robots have saved workers from tedious and dull assembly line jobs, and increased production and savings in the process. But, what's easy for a human assembler can be difficult or impossible for a robot.

To ensure success with robotic assembly, engineers must adapt their parts, products and processes to the unique requirements of the robot.
Those that handle tools and those that handle work can differentiate industrial robots. When equipped with gripper arms or tool changers, they can serve both functions.

Assembly automation with robots aims to reduce cost and increase the quality and efficiency of the operation. In environments hazardous to humans, having robots perform assembly tasks could save human lives.

Assembly has long been not only an important but also one of the most challenging applications for robotics.
There are many significant research issues related to the broad scope of assembly automation, from design for assembly, to tolerance analysis, assembly sequence planning, fixture design, etc. This paper is only focused on the issue of robotic motion for assembly in a virtual environment.

The principal target for assembly automation with robots will be applications involving high demands on flexibility. The flexibility and the reprogramming ability of robots will contribute to their expanded use in assembly operations. The robots are flexible in the sense that they can be programmed to assemble different products.

Robots are already being used in the manufacturing industry for parts handling, component insertion, assembly, and inspection when required, a high degree of repeatability.

The robot should be able to pick up a part and insert it without any further manipulation. The parts should have self-aligning features, such as lips or chamfers, to help the robot insert them.

An informal analysis of manufacturing engineers in the automatic assembly indicated that the most remarkable applications for robots in automatic assembly are given by the capabilities of today's robots and the maturity of the off-line programming software.

Studying the virtual robot systems and

detecting the movement of part assembly in the virtual environment and transforming this movement into symbolic language will sustain the control decisions making.

With these conclusions in mind, we next concentrate on several issues associated to using virtual robots for optimize of the automatic assembly process.

Automatic assembly is a computerized production control technique used in the production of manufactured goods to balance output of production with demand.

Robotic automatic assembly offers many important features and advantages that are not achieved with traditional fabrication techniques. These features include inserting, pressing, rolling and consolidation of the manipulated object, all in the automatic mode, precise control of object placement and orientation. Furthermore, the use of a robot manipulator increases the flexibility of the pieces placement process and allows for the fabrication of more complex structures.

Compared with other operations in industrial manufacture, the application of robotics to assembling operations is the area where the biggest potential for the robots' utilize is seen to be more exploited [1].

Automation of assembly can only take place through more flexible assembly systems [2]. More flexible assembly systems are needed to preserve the existing high level of automation in high-volume production over the long term. In this connection, high hopes are placed in assembly robots as the principal element in new flexible assembly systems.

## 2. Assembly Motion

An *assembly task* defines the process of putting together manufactured parts to make a complete product. It is a major operation in the manufacturing process of any product.

The concerned *assembly motion* is that of a robot manipulator holding a part and moving it to reach a certain a*ssembled state*, i.e., a required spatial arrangement or contact against another part. The main difficulty of assembly motion is due to the requirement for high precision or low tolerance between the parts in an assembled state. As a result, the assembly motion has to overcome uncertainty to be successful. Assembly motion strategies can incorporate compliant motion.

*Compliant motion* is defined as motion constrained by the contact between the held part and another part in the environment. As it reduces uncertainty through reducing the degrees of freedom (DOFs) of the held part, compliant motion is desirable in assembly. Therefore, a successful assembly motion has to move the peg out of such an unintended contact situation and lead it to reach the desired assembled state eventually. To make this transition, compliant motion is preferred. Often a sequence of contact transitions via compliant motion is necessary before the desired assembled state can be reached.

Assembly motion strategies that incorporate compliant motion can be broadly classified into two groups: *passive compliance* and *active compliant motion*, and both groups of strategies require certain information characterizing *topological contact states* between parts. Often a set of contact configurations share the same high-level contact characteristics.

Such a description is often what really matters in assembly motion as it characterizes a spatial arrangement that could be either an assembled state or just a *contact state* between a part and another part.

For contacting polyhedral objects, it is common to describe a *contact state* topologically as a set of *primitive contacts*, each of which is defined by a pair of contacting surface elements in terms of faces, edges, and vertices.

From the viewpoint of contact identification via sensing, however, both representations can result in states that are different by definition but indistinguishable in identification due to uncertainties.

*Passive compliance* refers to strategies that incorporate compliant motion for error correction during the assembly motion without requiring active and explicit recognition and reasoning of contact states between parts.

## 3. Assembly methods

There are many assembly methods for peg-in-hole insertion who leads to reduction and hopefully eventual correction of small position or orientation errors of the held part.

### 3.1 Remote center compliance

A *remote center compliance* (RCC) method was developed to assist high-precision peg-in-hole insertion [3]. The RCC is a mechanical spring structure used as a tool attached to the end-effector of a robot manipulator to hold a peg when it is inserted into a hole. The RCC is designed to have high stiffness along the direction of insertion but high lateral and angular compliances, and it projects the center of compliance near the tip of the peg (hence the name remote center compliance) to overcome small lateral and angular errors in the peg's position and orientation in response to the contact forces applied to the peg by the hole during insertion.

**3.2 Admittance Matrix**

As an alternative method to the RCC, a particular form of manipulator force control, damping control, was proposed to achieve compliant motion of the held part by the manipulator and to correct small location errors of the held part during assembly. This approach eliminates the need to build a mechanical device like an RCC to achieve error correction.

Among the force control laws [4], damping control is a common strategy, where a commanded velocity of the held part is modified based on the sensed force caused by contact between the held part and the environment. The resulted actual velocity leads to reduction and hopefully eventual correction of small position or orientation errors of the held part.

Let $v$ be a six-dimensional vector representing the actual translational and angular velocity of the held part, $v_0$ be the six-dimensional commanded velocity, and $f$ be a six dimensional vector representing the sensed force and moment. A linear damping control law is described as:

$$v = v_0 + A \cdot f \quad (1)$$

where, $\boldsymbol{A}$ is a 6×6 matrix, called an *admittance matrix* or accommodation matrix.

The effectiveness of such a damping control law depends on the existence and finding of a proper admittance matrix $\boldsymbol{A}$.

There are many researches on the design of a single $\boldsymbol{A}$ that can make an assembly operation successful regardless of what contact states the held peg may encounter in the process [5, 6, 8]. This is aimed at cases where a single commanded velocity would be sufficient to achieve an assembly operation when there were no uncertainty or error, such as certain peg-in-hole insertion operations.

One main approach to design $\boldsymbol{A}$ is based on explicit kinematics and static analysis of contact conditions under all possible contact states and mating requirements, which result in a set of linear inequalities as constraints on $\boldsymbol{A}$.

Learning is also used [5] to obtain an $\boldsymbol{A}$ that minimizes the force $\boldsymbol{f}$ without causing instability. Another approach applies perturbations to the end-effector during insertion in order to obtain richer force information.

**3.3 Learning Control for Assembly**

Another category of approaches is to learn proper control for a particular assembly operation through stochastic or neural-network-based methods.

The essence of most of these approaches is to learn to map a reaction force upon the held object, caused by contact to the next commanded velocity in order to reduce errors and to achieve an assembly operation successfully. An important approach [6] maps fused sensory data of pose and vision obtained during human demonstration of assembly tasks to compliant motion signals for successful assembly.

A different approach observes assembly tasks performed by human operators through vision or in a virtual environment [7] and generates a motion strategy necessary for the success of the task that consists of a sequence of recognized contact state transitions and associated motion parameters.

As a sequence of commanded velocities can be generated, unlike RCC or strategies based on a single admittance matrix described above, can be applied to cases with large uncertainties. However, the learned controllers are task dependent.

None of the above assembly motion strategies requires explicit recognition of contact states during their execution.

## 4. Constraint-based manipulations

For every object in the virtual environment, such as a feature element, a feature and a part, an event list is regarded as the attribute of this object and is attached to this object. An action list is connected to every event in the event list of the object.

On the base of this list, in this paper, are created, in a virtual environment, the virtual objects by means of the functions and procedures writhed in Delphi language. This action list shows the actions that will be done as soon as the event occurs. The constraint-based manipulations are realized by a basic interactive event and the actions being performed when these event occur. A basic interactive event is attached to every object.

Example for the basic interactive events is the grasping event, the moving event and the dropping event.

The framework of constraint-based manipulations for the grasping event is shown in figure 1.

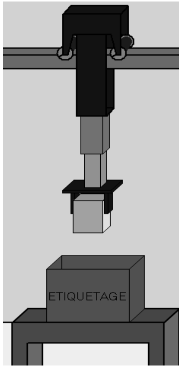


Figure 1. Virtual structure for constraint-based manipulations for the grasping event

The grasping event has an action for acquiring the current allowable motions of an object that is attached to it. An action for recognizing the constraints between objects is attached to the moving event and the dropping event.

As soon as the user grasps an object, the grasping event occurs and the current allowable motions of this object are derived from the hierarchically structured and constraint-based data model through constraint solving.

The constraint-based manipulations are acquired by constraining the motions of 3D hands to the allowable motions. This is done by transferring 3D motion data from the 3D input devices into the allowable motions of the object.

The constraint-based manipulations not only ensure that the precise positions of an object can be obtained, but also guarantee that the existing constraints will not be violated during the future operations.

Once a constraint is recognized during the constraint recognition, it will be highlighted and will await the user's confirmation.

Once it is confirmed, the recognized constraint will be precisely satisfied under the current allowable motions of the object and be inserted into the constraint-based data model. The satisfied constraint will further restrict the subsequent motions of the object.

**4.1. Representation of allowable motions**

The constraints between objects are implicitly created by the constraint-based manipulations with automatic constraint recognition and precise constraint satisfaction [8].

The newly created constraint reduces the DOFs of the object being manipulated and implicitly provides a confinement to the future operations applied to the object.

The remaining DOFs define the allowable motions of the object. The allowable motions explicitly describe the next possible operations and ensure that future operations will not violate the existing constraints.

The allowable motions are represented as a mathematical matrix for conveniently deriving the allowable motions of an object from the constraints applied to this object.

For every object in a free space, its configuration space has six DOFs: three translational DOFs and three rotational DOFs. To simplify the computation and to clarify the presentation of the allowable motions, we divide the configuration space along three linearly independent directions: X-axis, Y-axis and Z-axis. Therefore, some basic DOFs, i.e. three translational DOFs and three rotational DOFs can be obtained. Furthermore, the three basic translational or rotational DOFs are linearly independent among each other.

Any remaining DOF used to define the allowable motions can be described by these basic DOFs, therefore the allowable motions can be represented by these basic DOFs, as the following matrix:

$$\begin{bmatrix} T_x & R_x & T_{x_{\min}} & T_{x_{\max}} & R_{x_{\min}} & R_{x_{\max}} \\ T_y & R_y & T_{y_{\min}} & T_{y_{\max}} & R_{y_{\min}} & R_{y_{\max}} \\ T_z & R_z & T_{z_{\min}} & T_{z_{\max}} & R_{z_{\min}} & R_{z_{\max}} \end{bmatrix} \quad (2)$$

In this matrix the first column elements $T_x$; $T_y$ and $T_z$ are the linear translations along X-axis, Y-axis and Z-axis, respectively, and the second column elements $R_x$; $R_y$ and $R_z$ are the rotations about the corresponding axis, respectively. The values of these elements in the matrix are either 0 or 1. Integer 1 indicates that the motion is allowed in the direction along the corresponding axis. Integer 0 indicates that the motion is not allowed in the corresponding axial direction.

The third and fourth column elements are the allowable ranges of the three translations, which are defined by the minimum and maximum values of the three translations. For example, $T_{xmin}$ and $T_{xmax}$ are the minimum and maximum values of the translation along X-axis. The fifth and sixth column elements are the allowable ranges of the three rotations, which are defined by the minimum and maximum values of the three rotations. For example, $R_{xmin}$ and $R_{xmax}$ are the minimum and maximum values of the rotation about the X-axis. If the translation or rotation along some axis is not allowed, the corresponding minimum and maximum values are zero.

### 4.2. Constraint solving for deriving allowable motions

Since most constraints are geometric constraints and they are shown as the limitation of relative geometric displacements between objects, i.e. the limitation of DOFs, the constraints applied to an object can be mapped to the DOFs of this object. In fact, the relationship from constraints to DOFs can be extended to the relationship from a set of constraints to the combination of DOFs. Therefore, the representation of constraints can be obtained by analyzing and reasoning the DOFs of an object, and constraint solving can also be regarded as a process of analyzing and reasoning the DOFs of an object. Based on this, a procedure-based DOF combination method occurs for solving 3D constraints. This method combines DOF analysis with 3D direct manipulations in the virtual environment and has an intuitive solving way.

According this procedure, the current allowable motions of an object are derived from the current remaining DOFs of the object.

The action of grasping an object is interpreted by the constraint solver as requesting the current remaining DOFs of the object. The current constraints applied to the object can be obtained from the hierarchically structured and constraint-based data model.

Initially, the object is unconstrained and has six remaining DOFs. If there is only one constraint applied to the object, the current remaining DOFs can be directly obtained by DOF analysis. If there are multi-constraints (more than one) applied to the object, the current remaining DOFs of the object can be obtained by DOF combination.

The DOF combination for solving multi-constraints is based on the DOF analysis for solving individual constraints. Within the limitation of the current remaining DOFs determined by the current constraints, the object aims at satisfying a new constraint recognized by the current constraint-based manipulations applied to the object.

The new constraint is precisely satisfied under the current allowable motions of the object and is subsequently inserted in the constraint-based data model to update the current constraints applied to the object. The update of the current constraints results in the update of the current remaining DOFs of the object, and thus results in the update of the current allowable motions of the object.

Since DOFs are divided into three basic translational DOFs and three basic rotational DOFs, it is easy to connect a constraint with remaining DOFs by analyzing the remaining basic translational and rotational DOFs corresponding to the constraint.

For example, if a top of box is placed on a box (figure 2) and they are also needed to be lateral faces-aligned, the constraints between these two corps are the ‘‘against’’ and ‘‘line-alignment’’ constraints.

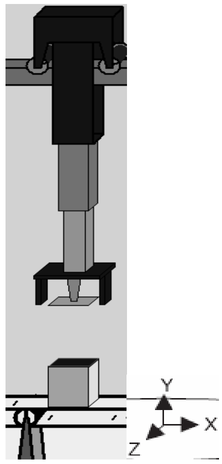


Figure 2. Lateral faces-aligned of two corps

By DOF analysis, the top of box has the translational DOFs $T_x$, $T_z$ and the rotational DOF $R_y$ for the ‘‘against’’ constraint, and the translational DOF $T_y$ and the rotational DOF $R_y$ for the ‘‘line-alignment’’ constraint. Similarly, the allowable motions matrices that correspond to other individual constraints can also be obtained by DOF analysis.

The DOF combination is used to represent the remaining DOFs that correspond to multi-constraints. It refers to the intersection within DOF of the allowable motions that respectively correspond to individual constraints.

Therefore, the DOF combination can be regarded as the individual combinations of the six translational and rotational DOFs, and can be further represented as the combination of the allowable motion matrices that respectively correspond to individual constraints. In such a way, the remaining DOFs of an object, that correspond to multi-constraints, can be obtained and the allowable motion matrix that corresponds to multi-constraints can also be acquired.

## 5. Implementation and results

A prototype system [9] for intuitive and precise solid modeling in a virtual environment through constraint based 3D direct manipulations has been implemented on the Delphi platform with Reality graphics workstation. The system components are shown in figure 3.

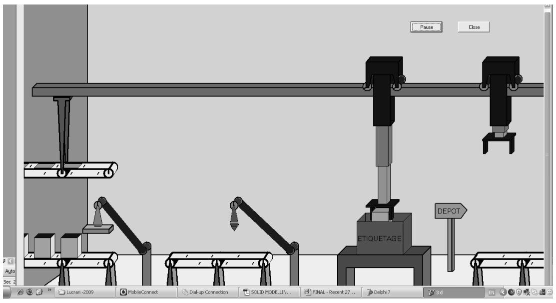


Figure 3. Virtual flexible line for assembly-case study

The body actor communicates with other actors and handles all aspects of user interaction. It receives and processes the information from the input actor. It also monitors and processes the events and actions in the virtual environment and outputs the processed results to the visual actor.

The collision actor resides in the system to detect the possible collisions between the virtual objects in the virtual environment.

The informatics modeler is in charge of all aspects of modeling and simulation, to establish the hierarchically structured and constraint-based data model.

The system framework is illustrated in figure 3. It consists of three modules, i.e. the hierarchically structured and constraint-based data model, the constraint processing and the modeling process.

During the modeling process, parts are created from feature primitives by constraint-based manipulations through locating feature primitives.

A feature library for providing some basic primitives is developed to support solid modeling. It is also employed to support polygon modeling through its triangulation function.

The hierarchically structured and constraint-based data model represents the entire solid modeling process with various design levels and the constraints at the different levels. It also provides the constraints to generate precise constraint-based manipulations.

## 6. Conclusions

Simulation planning processes simulation at virtual prototype level, have been established to allow planning of the motion control system.

In this paper, the authors use a case study based on a simplified assembly line realized in Delphi programming environment.

The paper describes the adopted solutions used to perform the constraint-based manipulations tasks.

## References


1. Reinhart, G., Werner, J.: *Flexible automation for the assembly in motion*. General Assembly of College International pour la Recherche en Productique (CIRP)- Journal of Manufacturing Science and Technology, p. 25-28, vol.56, No.1, May 2007, ISSN 0007-8506, Dresden, Germany
2. Hu, X. and Zeigler, B.P.: *Model Continuity to Support Software Development for Distributed Robotic Systems: a Team Formation Example*. Journal of Intelligent & Robotic Systems, Theory & Application, Vol. 39, Issue 1, January 2004, p. 71-87, ISSN 0921-0296, Springer Netherlands
3. Kim J, Kim K, Choi K, Lee J.Y.: *Solving 3D geometric constraints for assembly modeling*. International Journal Advanced Manufacturer Technologies, Vol. 16, No 11, September 2000, p. 843-849, ISSN 0268-3768, Springer Netherlands.
4. Li, Y.T., Hu, S.M., Sun, J.G.: *A constructive approach to solving 3-D geometric constraint systems using dependence analysis*. Computer Aided Design Journal, p. Vol. 34, Issue 2, February 2002, 97-108, ISSN 0010-4485, Elsevier
5. Balazs, A. et al: *Fat borders: gap filling for efficient view-dependent LOD NURBS rendering*. Computer Graphic Journal, Vol. 28, Issue 1, February 2004, p.79–85, ISSN 0097-8493, Elsevier
6. Floriani, L.D., Magillo, P., Puppo, E., Sobrero, D.: *A multi - resolution topological representation for non-manifold meshes.* Computer Aided Design Journal, Vol. 36, Issue 2, June, 2004, p. 141–59, ISBN 1-58113-506-8, Saarbrücken, Germany
7. Fratu, A. et al: *Industrial Robots in a Flexible Assembly System*. RECENT Journal, Vol.10, No.1 (25), p. 28-31, ISSN 1582-0246, Brasov, Romania
8. Zhong, Y. et al: *A methodology for solid modeling in a virtual reality environment.* International Journal of Robotics and Computer-Integrated Manufacturing, Vol.21, Issue 6, December 2005, p. 528-549, 0736-5845, Pergamon
9. Choi, S.H, Chan, A.M.M.: *A virtual prototyping system for rapid product development.* Computer Aided Design Journal, Vol. 36, Issue 5, April 2004, p. 401-412, ISBN 1-58113-506-8, Saarbrücken, Germany